1990

A dual ascent algorithm for traffic assignment problems

Hearn, Donald W.

Transpt. Res. -B, Vol. 24B, No. 6, pp. 423-430. 1990.
http://hdl.handle.net/10945/43130

Transpn. Res.-B, Vol. 24B, No. 6, pp. 423–430, 1990
Printed in Great Britain.

0191-2615/90 $3.00+.00

A DUAL ASCENT ALGORITHM FOR TRAFFIC ASSIGNMENT PROBLEMS

DONALD W. HEARN
ISE Department, 303 Weil Hall, University of Florida, Gainesville, FL 32611, U.S.A.

and

SIRIPHONG LAWPHONGPANICH
Operations Research Department, Code 55, Naval Postgraduate School, Monterey, CA 93943, U.S.A.

Abstract—A dual decomposition algorithm is developed for large-scale traffic assignment problems. In contrast to standard methods, this algorithm does not require that the system or user optimal objective function be differentiable and it allows bounds on the arc flows. Iterates alternate between dual ascent steps and calculations of shortest paths as in the Frank-Wolfe method. Although a dual method, it produces feasible flow patterns at each iteration. Convergence of the method is proven and a computational example is given.

1. INTRODUCTION

In the standard traffic assignment problem (TAP) (e.g. LeBlanc et al., 1974; Nguyen, 1976; Steenbrink, 1974), the capacities on arcs are often treated implicitly in the delay formula. The version of the TAP considered here treats the capacities explicitly as constraints and can be considered as a special case of the nonlinear multicommodity flow problem (MCFP). As in the standard TAP, a commodity represents travel demands with a common origin. Mathematically, the nonlinear MCFP can be stated as follows:

$$P: \quad f^* = \min_{(x^c, z)} \quad \Sigma_a f_a(z_a)$$

$$\text{s.t.} \quad Ax^c = b^c \quad \text{for } c = 1,2, \ldots C$$

$$x^c \geq 0 \quad \text{for } c = 1,2, \ldots C$$

$$\Sigma_c x_a^c = z_a \quad \text{for all arcs } a$$

$$0 \leq z_a \leq p_a \quad \text{for all arcs } a$$

where C represents the number of distinct origins or commodities, A is the node-arc incidence matrix for the street network, b^c is the travel demand vector for commodity c, x_a^c is the flow for commodity c on arc a, z_a is the total flow on arc a, p_a is the capacity on arc a, and f^* denotes the optimal objective value. The congestion function on arc a, $f_a(z_a)$, is assumed to be an increasing convex function of z_a. With this assumption, the penultimate constraint which 'couples' together the flows of every commodity may be written as

$$\Sigma_c x_a^c \leq z_a \quad \text{for all arcs } a$$

because the minimization will force this constraint to be equality at the solution.

Applying the Frank-Wolfe (FW) algorithm (Frank and Wolfe, 1956) to the nonlinear MCFP would not be effective because it would require solving a linear MCFP as a subproblem every iteration. With an additional assumption that the congestion function $f_a(z_a)$ tends to infinity as z_a approaches its capacity p_a, Daganzo (1977a,b) (see also Hearn and Ribera, 1981) presents a modification of FW to address the nonlinear MCFP without having to solve the linear MCFP subproblem. However, Boyce et al. (1981) found that a congestion function satisfying the additional assumption, in particular the Davidson (1966) congestion function, empirically leads to unrealistically high travel times and devious rerouting of trips. They caution that the resulting flow assignment should be used with extreme caution in any planning application.

As an alternative, this paper considers a dual ascent technique for solving the nonlinear MCFP which does not require that the congestion function tends to infinity as the total arc flow approaches the capacity. However, it still retains a similar subproblem structure as in the modification of the FW algorithm in that evaluating the dual function is equivalent to solving shortest path and simple one variable minimization problems. The main result of this paper is the convergence of this dual ascent technique. To demonstrate the relative effectiveness of the algorithm an example problem is solved.

The remaining sections are as follows. The dual ascent technique is presented in Section 2. Section 3 provides the proof of convergence. Finally, Section 4 presents the computational results on an example problem.

2. A DUAL ASCENT ALGORITHM FOR THE NONLINEAR MCFP

A relaxation of the nonlinear MCFP is obtained by associating multipliers $v \geq 0$ with the coupling constraint. The resulting (partial) dual problem is

$$(D) \quad L^* = \max_{v \geq 0} \quad L(v)$$

where

$$L(v) = \Sigma_a \min_{0 \leq z_a \leq p_a} \{f_a(z_a) - v_a z_a\} + \Sigma_c \min \{vx^c : Ax^c = b^c, x^c \geq 0\}.$$

Note that the evaluation of $L(v)$ is separable in z_a and x^c. The first minimization is accomplished by one-variable minimization, often resulting in a closed form expression for the minimizing z_a as a function of v_a. So long as this can be carried out efficiently, there is no requirement that $f_a(z_a)$ be differentiable. The second is accomplished by solving C shortest path problems with arc lengths defined by v.

The objective of the dual problem, $L(v)$, is concave and has subgradients at every $v \geq 0$. Algorithms to solve dual problems often are of the cutting plane variety—every evaluation of $L(v)$ provides a tangent plane (a cut) which overestimates $L(v)$. A linear program maximizes the minimum of the cuts and provides a new upper bound on $L(v)$ as well as a new value of v. Theoretical convergence results exist for such methods, but performance is often poor. For example, one drawback is that the dual objective does not increase monotonically and another is that the rate of convergence can be slow.

Below, we give a new cutting plane algorithm for (D) which has a dual ascent line search step. For conciseness we will assume that the nonlinear MCFP has been recast in the notationally compact form of a nonlinear program:

$$\begin{aligned} (\text{NLP}) \quad & \min_{x} \quad f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \\ & x \in X \end{aligned}$$

where x is the vector of all variables $(x^1, \ldots, x^C, z)$. The set X is defined by the conditions $Ax^c = b^c$, $x_a^c \geq 0$ and $0 \leq z_a \leq p_a$, and the condition $\Sigma_c x_a^c \leq z_a$ becomes $g(x) \leq 0$.

A DUAL ASCENT ALGORITHM

Step 0. Find a point $x_0 \in X$ such that $g(x_0) < 0$. Set $v_0 = 0$ and $k = 1$.

Step 1. Let (w_k, u_k) solve the (master) problem:

$$\begin{aligned} (\text{M1}) \quad & \max_{(w,u)} \quad w \\ & w \leq f(x_i) + u\, g(x_i) \quad i = 0, \ldots, k-1 \\ & u \geq 0 \end{aligned}$$

Step 2. Set $d_k = u_k - v_{k-1}$. If $k = 1$, set $t_k = 1$. Otherwise, let

$$t_{\max} = \text{arg. max}\{L(v_{k-1} + td_k) : 0 \le t \le t_{up}\}$$

$$\text{where } t_{up} = \max\{t : v_{k-1} + td_k \ge 0\}.$$

If $t_{\max} \le 1$, then pick *any* nonzero $t_k \in [t_{\max}, 1]$. Otherwise, pick *any* $t_k \in [1, t_{\max}]$. Set $v_k = v_{k-1} + t_k d_k$ and solve the (sub)problem

(S1) $$L(v_k) = \min\{f(x) + v_k g(x) : x \in X\}.$$

Let x_k be an optimal solution. If x_k is not unique, choose x_k so that

$$f(x_k) + u_k g(x_k) \le f(x_k) + v_k g(x_k) = L(v_k).$$

Step 3. If $w_k = L(v_k)$, stop. Otherwise, set $k = k + 1$ and go to step 1.

In the standard cutting plane algorithm, the solutions u_k to the master problem (M1) are considered to be the iterates (solutions) generated by the algorithmic process, and the normal stopping criterion would be to test whether $w_k = L(u_k)$ at the end of Step 1. However, the above algorithm uses u_k to form a search direction (i.e. $d_k = u_k - v_{k-1}$), and treats v_k as the iterates instead. Note that problem (M1) is a linear program and the subproblem (S1) evaluates $L(v_k)$.

For the traffic assignment or multicommodity flow problem, solving problem (S1) is equivalent to solving (i) C independent shortest path problems with arc costs defined by v_k and (ii) a set of one variable minimization problems. These one variable problems, as described earlier, generally result in closed form expressions. The first part of Step 2 is a line search in the direction d_k. Hearn and Lawphongpanich (1989) show that this direction ascends the dual function, $L(v)$, whenever it is differentiable at v_{k-1}. Setting $t_k = t_{\max}$ places v_k at the maximizing point. Setting $t_k = 1$ places v_k at u_k, and the algorithm becomes the standard dual cutting plane method (see, e.g., Dantzig, 1963; Eaves and Zangwill, 1971; Magnanti *et al.*, 1976) which is convergent but does not ascend the dual. The specific choice of t_k is *inexact*, however, to allow for inaccuracy in the line search and the convergence result of the next section holds for any choice of t_k in the range(s) given. The choice of x_k in Step 3 is important for convergence, and Lemma 2 of the next section guarantees that the condition on x_k can always be met.

3. VALIDATION AND CONVERGENCE

Theorem 1 below shows that, if Step 1 is allowed to execute after the stopping criterion in Step 3 is satisfied, the solution to (M1) would satisfy the stopping criterion of the standard cutting plane algorithm mentioned above. This therefore validates the stopping rule in Step 3. Lemma 2 then justifies the requirement for x_k in Step 2. Lemma 3 and Theorem 4 establish the convergence of the algorithm. Finally, Theorem 5 demonstrates how to obtain a primal solution.

Theorem 1: If $L(v_k) = w_k$, then v_k is an optimal solution.

Proof: Let x_k be a solution to the subproblem

$$\min\{f(x) + v_k g(x) : x \in X\}$$

Then, $L(v_k) = w_k$ implies that

$$w_k = f(x_k) + v_k g(x_k). \tag{1}$$

However, since x_k solves the subproblem,

$$w_k = f(x_k) + v_k g(x_k) \le f(x_i) + v_k g(x_i) \quad i = 0, \ldots, k-1. \tag{2}$$

From (1) and (2) we have that (w_k, v_k) is feasible to the master problem at step $k + 1$ ($M1^{k+1}$), i.e.,

$$\begin{aligned} \max \quad & w \\ \text{s.t.} \quad & w \le f(x_i) + u\, g(x_i) \quad i = 0, \ldots, k. \\ & u \ge 0. \end{aligned}$$

Note that (w_k, u_k) is an optimal solution to the master problem at Step k ($M1^k$) which has one less constraint than $M1^{k+1}$. Since (w_k, v_k) is feasible to $M1^{k+1}$ and has the same objective value as the optimal solution to $M1^k$, (w_k, v_k) must be optimal to $M1^{k+1}$. However, $L(v_k) = w_k$ implies that v_k is an optimal dual solution.

Lemma 2: In Step 3 there exists a solution to the (sub)problem:

$$L(v_k) = \min \{f(x) + v_k g(x) : x \in X\}$$

such that

$$f(x_k) + u_k g(x_k) \leq f(x_k) + v_k g(x_k).$$

Proof: If $t_{\max} = 1$, then the theorem is immediate. Assume that $t_{\max} > 1$. Denote $v_{\max} = v_{k-1} + t_{\max} d_k$. Then, $v_k = \beta v_{\max} + (1 - \beta) u_k$ for some $\beta \in (0,1)$ and by concavity of $L(u)$

$$L(v_k) \geq \beta L(v_{\max}) + (1 - \beta) L(u_k) \geq L(u_k). \tag{3}$$

Let $X(v_k)$ denote the (compact) set of solutions to the (sub)problem. To obtain a contradiction, assume that

$$f(x) + u_k g(x) > f(x) + v_k g(x) \qquad \text{for all } x \in X(v_k).$$

Then

$$(u_k - v_k) g(x) > 0 \qquad \text{for all } x \in X(v_k)$$

which implies

$$\min \{(u_k - v_k) g(x) : x \in X(v_k)\} > 0.$$

Thus the directional derivative of $L(v)$ at v_k in the direction $(u_k - v_k)$ is positive (i.e. $(u_k - v_k)$ is an ascent direction). However, v_k is a point on the line connecting $v_{\max}$ to u_k and by concavity of $L(v)$ moving toward u_k must decrease its value. This contradicts the statement that $(u_k - v_k)$ is an ascent direction. (The case $t_{\max} < 1$ is proved similarly.)

Lemma 3: At iteration k of the algorithm

(1) $w_{k-1} \geq w_k \geq L^*$

(2) If $L(v_k) = w_k$, then $L(v_k) = L^*$; that is, v_k is an optimal dual solution.

Proof: (1) follows from Lemma 3.1 of Magnanti *et al.* (1976) and (2) follows from Theorem 1.

Theorem 4: Assume that there exist a point $x_0 \in X$ such that $g(x_0) < 0$. If the dual ascent algorithm generates a sequence $\{v_k\}$, there exists an index set $K \subseteq \{1,2, \ldots\}$ such that

(1) $v^\infty = \lim_{k \in K} v_k$ is optimal to the dual problem, and

(2) $\lim_{k \in K} w_k = w^\infty = L^*$.

Proof: Under the assumption there exists a index set K such that the subsequence $\{u_k\}_{k \in K}$ converges (see Magnanti *et al.*, 1976). Let $u^\infty = \lim_{k \in K} u_k$.

Since (w_k, u_k) solves the kth master problem, we have

$$f(x_j) + u_k g(x_j) \geq w_k \geq L^* \quad \text{for } j = 0, \ldots, k - 1 \tag{4}$$

where the right inequality follows from Lemma 3. Note that w^∞ exists because the w_k are monotonically decreasing and bounded below by L^*. Taking the limit in eqn (4) for $k \in K$, we obtain

$$f(x_j) + u^\infty g(x_j) \geq w^\infty \geq L^* \quad \text{for } j = 0,1,2 \ldots \tag{5}$$

By assumption, $g(x)$ is a continuous function mapping R^n into R^m and X is compact, there is a real number β such that $|g(x)| \leq \beta$ for all $x \in X$. Then,

$$|f(x_k) + u_k g(x_k) - f(x_k) - u^\infty g(x_k)| = |(u_k - u^\infty) g(x_k)| \leq \beta|(u_k - u^\infty)|.$$

Consequently, for any given $\epsilon > 0$ there is a $k_1 \in K$ such that for all $k \in K$ and $k \geq k_1$ the last term is bounded by ϵ. Therefore,

$$|f(x_k) + u_k g(x_k) - f(x_k) - u^\infty g(x_k)| \leq \epsilon$$

$$\epsilon \geq f(x_k) + u_k g(x_k) - f(x_k) - u^\infty g(x_k) \geq -\epsilon.$$

Examining the second inequality, we obtain

$$f(x_k) + u_k g(x_k) \geq f(x_k) + u^\infty g(x_k) - \epsilon,$$

and from Lemma 2, we have

$$L(v_k) = f(x_k) + v_k g(x_k) \geq f(x_k) + u_k g(x_k) \geq f(x_k) + u^\infty g(x_k) - \epsilon.$$

From (5) and the definition of L^*

$$L^* \leq w^\infty \leq f(x_k) + u^\infty g(x_k) \leq L(v_k) + \epsilon \leq L^* + \epsilon.$$

Since $\epsilon > 0$ is arbitrary, $w^\infty = L^*$. Furthermore

$$\lim_{k \in K} L(v_k) = L(\lim_{k \in K} v_k) = L(v^\infty) = L^*$$

where the first equality follows from the continuity of L.

Lemma 5: Let λ_i^k, $i = 0, \ldots, k-1$ be the optimal linear programming dual variables associated with (w_k, u_k) in Step 1 at iteration k. Then

$$y^k = \sum_{i=0}^{k-1} \lambda_i^k x_i$$

is a feasible solution to (NLP). If, in addition

$$\sum_{i=0}^{k-1} \lambda_i^k f(x_i) - L(u_k) \leq \epsilon$$

then $f(y^k) \leq f^* + \epsilon$.

Proof: This result follows from the convexity of f and g in the problem (NLP) by using the fact that $\sum_{i=0}^{k-1} \lambda_i^k = 1$, $\lambda_i^k \geq 0$ for all k.

4. COMPUTATIONAL EXAMPLE

In the last section, the convergence of the cutting plane algorithm with the addition of a linear search step is demonstrated. To examine the effect of the line search, we solved a test problem from Goffin (1987) with 22 arcs 14 nodes, 23-OD pairs, and 5 origins. An (NLP) formulation of this problem would have 93 rows, 22 nonlinear variables, 120 linear variables, and 352 nonzero elements. The volume delay formula is

$$f_a(z_a) = \frac{z_a}{c_a - z_a}$$

for every arc a.

Our implementation of the method relaxes two requirements of the stated algorithm: the calculation of $t_{\max}$ is approximated to prevent excessive time in the line search and no

Table 1. Dual algorithm results without line search ($t_k = 1$)
(standard dual cutting plane algorithm)

Iter.	Pivot	w_k	Dual	Err	Gap	cpu.
1	1	185.7116	6.6065	93.61	2354.61	0.026
2	2	185.0120	−108.4341	93.61	2345.42	0.055
3	3	183.8733	12.6054	87.81	1258.82	0.083
4	4	182.1552	−100.5754	87.81	1246.19	0.113
5	6	181.2106	−438.4205	87.81	1239.25	0.153
6	7	179.0065	−103.3858	87.81	1223.05	0.183
7	9	177.5494	−418.0671	87.81	1212.34	0.226
8	11	173.9477	−172.7024	87.81	1185.86	0.270
9	15	172.1916	−43.6436	87.81	1172.96	0.337
10	17	169.8475	21.8088	78.91	649.04	0.379
20	64	153.6691	−26.5477	72.91	433.00	1.398
30	180	138.0072	53.2345	25.04	77.04	3.825
40	319	123.2617	77.1200	25.04	58.26	6.818
50	472	112.9877	89.0747	11.63	23.38	10.304
60	557	109.8193	91.2045	7.22	14.31	12.442
70	668	108.1534	99.2914	3.99	8.84	15.318
80	786	106.8435	99.5316	2.58	6.00	18.459
90	895	105.8758	99.0826	2.40	4.85	21.392
100	1020	105.3563	100.9082	2.30	4.23	25.075
125	1259	104.3922	101.8204	0.63	1.57	32.612
150	1448	103.9188	102.5478	0.51	1.00	39.162
175	1686	103.6872	102.9460	0.30	0.56	48.130
200	1848	103.5468	103.1820	0.20	0.33	54.839
225	2005	103.4822	103.2331	0.08	0.15	61.817
238	2066	103.4617	103.3595	0.05	0.10	64.665

check is made to ensure that d_k is not a descent direction as may occur when the subproblem has multiple solutions. Thus, the ascent property of the method may not hold at every iteration. Instead we expect dual ascent sufficiently often to make the method more effective than the cutting plane method without the line search step.

Tables 1–3 display the results for the dual algorithm for three different values of t_k. In these three tables, the error and the gap values are calculated as follows:

$$\text{error} = 100 * (\text{opt. dual} - \text{best dual})/\text{opt. dual}$$
$$\text{gap} = 100 * (w_k - \text{best dual})/(\text{best dual} + 1).$$

Table 2. Dual algorithm results with $t_k = t_{\max}$ (approximately)

Iter.	Pivot	w_k	Dual	Err	Gap	cpu	Fun. Eval
1	1	185.7116	6.6065	93.61	2354.61	0.028	1
2	2	185.0120	8.8448	91.45	1798.44	0.190	9
3	3	183.8679	16.6492	83.90	947.46	0.387	19
4	5	182.4732	31.4760	69.56	464.85	0.519	25
5	5	182.3496	24.9551	69.56	464.57	0.568	26
6	8	182.0098	25.9035	69.56	463.52	0.779	35
7	12	181.2055	26.0612	69.56	461.05	1.022	45
8	14	180.5734	29.2225	69.56	459.10	1.239	55
9	18	179.1596	29.4240	69.56	454.75	1.418	61
10	23	175.4888	17.8790	69.56	443.44	1.640	62
20	81	153.1643	82.3477	20.37	84.97	4.188	135
30	209	133.1641	100.5707	2.75	32.09	8.850	229
40	292	115.3490	101.1016	1.69	13.33	12.063	297
50	379	104.5042	103.1863	0.22	1.27	15.905	384
60	486	103.8953	103.3528	0.06	0.52	20.451	474
70	597	103.6476	103.3793	0.03	0.26	25.073	563
80	715	103.5534	103.3980	0.01	0.15	30.159	650
86	809	103.5018	103.3994	0.01	0.01	33.848	699

Table 3. Dual algorithm results with heuristic choice of t_k

Iter.	Pivot	w_k	Dual	Err	Gap	cpu	Fun. Eval
1	1	185.7116	6.6065	93.61	2354.61	0.028	1
2	2	185.0120	8.8447	91.45	1789.44	0.195	9
3	3	183.9501	18.5167	82.09	847.65	0.393	19
4	5	182.0492	19.5644	81.08	790.13	0.605	29
5	6	181.8431	36.0554	65.13	393.43	0.741	35
6	8	181.1697	36.1486	65.04	390.38	0.953	45
7	11	179.5747	36.2575	64.94	384.67	1.149	53
8	13	178.7324	34.4356	64.94	382.41	1.213	54
9	17	173.4795	67.7107	34.52	153.93	1.342	57
10	23	171.2281	68.9878	33.29	146.08	1.731	67
20	110	151.5452	81.0819	21.59	85.85	5.212	159
30	218	135.9153	98.2684	4.97	37.92	9.466	255
40	320	115.8433	99.4474	3.83	15.32	13.440	339
50	398	105.9836	103.1386	0.26	2.73	17.287	436
60	548	104.1151	103.3619	0.05	0.72	22.917	530
70	649	103.5723	103.3903	0.02	0.18	27.509	627
78	721	103.5035	103.4022	0.01	0.01	30.966	694

The addition of 1 in the definition of the gap value is to prevent division by zero. For Tables 2 and 3, the FUN EVAL column gives the cumulative number of subproblems solved during the line search step.

Table 1 shows the results of the standard cutting plane algorithm where there is no line search (i.e. in Step 2 of the algorithm), $t_k = 1$ at each iteration. In Table 2, the results are for $t_k = t_{\max}$ (approximately). If the dual function at $t_{\max}$ is nondifferentiable there is no efficient method to choose x_k as required in Step 2. To avoid this, t_k is heuristically set to .99 $t_{\max}$ (if $t_{\max} > 1$) or 1.01 $t_{\max}$ (if $t_{\max} < 1$) and to .01 (if $t_{\max} \leq 0$) in Table 3. It is clear from these tables that the line search step improves the normal cutting plane method by reducing the cpu time on the order of 50%, with an additional 10% improvement if the heuristic choice is made for t_k.

Although the results reported are for only one example, they corroborate the results of the same technique applied to other types of problems in Hearn and Lawphongpanich (1989). Furthermore, improvements to the method are possible. For example the efficiency of the algorithm may be improved by (1) selecting a more efficient technique for solving the linear problem (M1) (e.g. see Goffin, 1987), (2) dropping the inactive cuts to reduce the size of (M1), and (3) solve (M1) approximately. We also note that the example problem has a special structure which permits efficient calculation of shortest paths. In general, we expect a trade-off between solving the master and the subproblem. When the subproblem is relatively easy to solve, it should be advantageous in terms of the overall cpu time to solve the subproblem many times in performing the line search. On the other hand if the subproblem is relatively difficult, fewer subproblems should be solved during the line search step.

The method of this paper may be contrasted with the dual approach of Fukushima (1984,a,b) for the MCFP. In (Fukushima, 1984a) he proposes a different cutting plane algorithm in which the master has a quadratic objective and a line search is performed only on those iterations satisfying certain criteria. This represents a different strategy of allocating cpu time to solving the master and subproblem.

Acknowledgement—The research was supported in part by NSF grants DDM-8814075 and DDM-8814499 and a grant from the Naval Postgraduate School direct funding program.

REFERENCES

Boyce D. E., Janson B. N., and Eash R. W. (1981) The effect on equilibrium trip assignment of different link congestion functions. *Transpn. Res.* **15B**, 223-232.

Daganzo C. F. (1977a) On the traffic assignment problem with flow dependent cost-I. *Transpn. Res.* **11**, 433-437.

Daganzo C. F. (1977b) On the traffic assignment problem with flow dependent cost-II. *Transpn. Res.* **11**, 439-441.

Dantzig G. B. (1963) *Linear Programming and Extensions*. Princeton University Press, Princeton, NJ.
Davidson K. B. (1966) A flow-travel time relationship for use in transportation planning. Proc. Aus. Road. Res. Board 3, 183-194.
Eaves C. B. and Zangwill W. I. (1971) Generalized cutting plane algorithms. *SIAM J. Control* **9**, 529-542.
Frank M. and Wolfe P. (1956) An algorithm for quadratic programming. *Naval Res. Logist. Quart.* **3**, 95-110.
Fukushima M. (1984a) A nonsmooth optimization approach to nonlinear multicommodity network flow problems. *Operations Research Society of Japan*, **27**(2) 151-177.
Fukushima M. (1984b) On the dual approach to the traffic assignment problem. *Transpn. Res.* **18B**,(3), 235-245.
Goffin J. L. (1987) Affine methods in nondifferentiable optimization. CORE Discussion Paper No. 8744, Center for Operations Research and Econometrics, University Catholique de Louvain, Louvain, Belgium, 1-24.
Hearn D. W. and Ribera J. (1981) Convergence of the Frank-Wolfe method for certain bounded variable traffic assignment problems. *Transpn. Res.* **15B**, 437-442.
Hearn D. W. and Lawphongpanich S. (1989) Lagrangian dual ascent by generalized linear programming. *Op. Res. Lettrs* **8**, 189-196.
LeBlanc L. J., Morlok E. K., and Pierskalla W. P. (1974) An accurate and efficient approach to equilibrium traffic assignment on congested networks. *Transportation Research Record 491*, Interactive Graphics and Transportation Systems Planning, 12-33.
Magnanti T. L., Shapiro J. F., and Wagner H. M. (1976) Generalized linear programming solves the dual. *Mgt. Sci.* **22**, 1195-1162.
Nguyen S. (1976) A unified approach to equilibrium methods of traffic assignment in traffic equilibrium methods. In *Lecture Notes in Economics and Mathemathical Systems, Vol. 118* (M. Florian ed.), Springer Verlag, Berlin, pp. 148-182.
Steenbrink P. A. (1974) *Optimization of Transport Networks*. Wiley, New York.